The Journal of Supercomputing (2022) 78:19020–19045
https://doi.org/10.1007/s11227-022-04629-7

A novel neural network training framework with data assimilation

Chong Chen[1] · Yixuan Dou[1] · Jie Chen[1] · Yaru Xue[1]

Accepted: 23 May 2022 / Published online: 15 June 2022

Abstract
In recent years, the prosperity of deep learning has revolutionized the Artificial Neural Networks. However, the dependence of gradients and the offline training mechanism in the learning algorithms prevents the Artificial Neural Networks from further improvement. In this study, a gradient-free training framework based on data assimilation is proposed to avoid the calculation of gradients. In data assimilation algorithms, the error covariance between the forecasts and observations is used to optimize the states. The Feedforward Neural Networks are trained by gradient decent, data assimilation algorithms (Ensemble Kalman Filter and Ensemble Smoother with Multiple Data Assimilation), respectively. Ensemble Smoother with Multiple Data Assimilation trains Feedforward Neural Networks with pre-defined iterations by updating the parameters (i.e. states) using all the available observations which can be regarded as offline learning. Ensemble Kalman Filter optimizes Feedforward Neural Networks when new observation available by updating parameters which can be regarded as real-time learning. Two synthetic cases with the regression of a Sine function and a Mexican Hat function are conducted to validate the effectiveness of the proposed framework. Quantitative comparison with the root mean square error and coefficient of determination show that better performance is obtained by the proposed framework than the gradient decent method. Furthermore, the uncertainty of the parameters is quantified which shows the reduction in uncertainty along with the iterations in Ensemble Smoother with Multiple Data Assimilation. The proposed framework explores alternatives for real-time/offline training the existing Artificial Neural Networks (e.g. Convolutional Neural Networks, Recurrent Neural Networks) without the dependence of gradients and conducting uncertainty analysis at the same time.

Keywords Neural networks · Training algorithm · Data assimilation · EnKF · ESMDA

✉ Chong Chen
chenchong@cup.edu.cn

Extended author information available on the last page of the article

1 Introduction

Artificial Neural Networks (ANNs) have been investigated and utilized extensively by researchers in numerous fields to conduct predictions and classifications based on the knowledge learning from training data [1, 2]. Significant accomplishments have been achieved by applying ANNs in computer vision, speech recognition and natural language processing [3, 4]. ANNs were mathematical models of biological neural networks which constitute animal brains [5] with neurons, connections (axons) and transfer functions (synapse). After decades of researches and developments, ANNs have evolved from Perceptron [6] to Hopfield network [7], to Backpropagation Neural Network [8] and more recently to deep learning [2] which promotes the third wave of Artificial Intelligence (AI). Nonlinear mapping capability was obtained by applying sufficiently large number of neurons, connections, weights, bias, transfer functions and learning algorithms. ANNs are capable of approximate any function with any given precision from a mathematical perspective [9, 10]. However, critical issues should be addressed for applying ANNs more effectively.

The first issue is the dependence of gradient during training ANNs. Although the number of neurons, connections, weights, bias, transfer functions are essential aspects for ANNs, a training procedure which adjusts the weights and biases is necessary to ensure the behaviour of ANNs as expected. Backpropagation has played an important role since 1980s which is efficient for training ANNs with a teacher-based supervised learning algorithm. The errors are backpropagated through the networks based on gradient decent algorithm (GD). However, the algorithm might be trapped in local minima because of the dependence of local gradient information. Although some improved methods (e.g. Batch Gradient Descent, Stochastic Gradient Descent and Mini-batch Gradient Descent) have been proposed, the convergence of ANNs during training stages is another problem which would further influence the performance of training and predicting. Therefore, some researchers propose to train ANNs with Heuristic Algorithms (HAs). For example, Zhao Hong proposed General Vector Machine (GVM) which trains ANNs with Monte Carlo algorithm and Least Square Errors [11]. The accuracy and generalizability performed well in relatively small data sets, but this method could hardly obtain satisfying results in large data sets with the exponential increase of computational cost. Simulated Annealing was integrated with GD and backpropagation to avoid local optima during training ANNs [12]. Researches and progresses have been obtained by applying Genetic Algorithm to adjust the weights and bias during training procedure. However, solid theoretical basis was missing due to the origination of HAs.

The second issue is the uncertainty analysis of ANNs. Uncertainty has always been intrinsic property in all kinds of prediction models (including black-box and deterministic models) [13]. Uncertainty analysis is to quantitatively identify and reduce uncertainties in models which tries to determine how likely the outputs are if some aspects of the system are not exactly known [14]. Ignoring the sources and influences of uncertainty would undermine the robustness and reliability of the results and analysis from a model [15, 16]. There are four important sources of uncertainties for a model [13]: (1) uncertainties in input data; (2) uncertainties in data used for calibration; (3) uncertainties in model parameters; (4) uncertainties due to imperfect model structure. Generally, there are broadly two groups of uncertainty assessment methods [17], i.e. the Bayesian methods [18] and the Generalized Likelihood Uncertainty Estimation (GLUE) method [19]. The Bayesian methods analyse the uncertainty of a model by assuming a prior probability over hypotheses to determine the probability of a particular hypothesis based on observed evidence. The GLUE method conducts a large number of model runs with many random parameter values selected from a priori probability distribution. The parameter values are accepted or discarded based on a certain subjective threshold which leads to a major drawback of GLUE method depending on subjective decisions rather than statistically consistent error models. Furthermore, the large number of model runs is not practical due to the computational cost of complex models.

Data Assimilation (DA) is originated from and has a long tradition in meteorology and oceanography [20, 21]. The essence of DA is to deal with uncertainty by assimilate different kinds of observations. It is well known that a free-running model will accumulate errors until its prediction is no long useful [22]. The only way to avoid this procedure is to allow the model to be influenced by observations [23]. DA provide a solution to evolve the models (update the states) by involving available observations. This procedure has different names in different fields, e.g. states estimation [24]; optimization [25]; history matching [26]; retrieval production [27]; inverse modelling [28]. The objective of DA is to produce information about the posterior Probability Density Function (PDF) by different approaches. There are three categories of Bayesian-based strategies of DA methods: (1) Variational DA with implementations of 3D-Var or 4D-Var; (2) Ensemble DA which implements based on Ensemble Kalman Filter (EnKF); (3) Monte Carlo methods which allow the assimilation of information with non-Gaussian errors. The EnKF [29] is derived from the merge of Kalman Filter [30] and Monte Carlo estimation methods [31]. The algorithm has been examined and applied in various fields such as metrology, oceanography, petroleum engineering and hydrogeology [32–34], since it was first proposed by Evensen [29]. The simple conceptual formulation and relative ease of implementation (no derivation of a tangent linear operator or adjoint equations are required) with affordable computational requirements result in the popularity of EnKF. The system states can be forecasted and updated simultaneously with minimized error covariance in real time. Bocquet et. al showed the possibilities of combining DA and machine learning from a Bayesian perspective [35]. In [36], the Kalman filter was used to sequentially update the output weights of a single-layer feedforward network based on Online Sequential Extreme Learning Machine to conduct online learning and handle the effects of multicollinearity. The Extended Kalman Filter (EKF) has been used to optimize parameters of Support Vector Machine [37], Feedforward Neural Network (FNN) [38–40], Radial Basis Function Neural network [41], Recurrent Neural Network (RNN) [42, 43]. Chen et. al proposed a training method based on the ensemble randomized maximum likelihood algorithm which avoided the gradient while training and performed uncertainty analysis at the same time [44]. Ensemble Smoother with Multiple Data Assimilation (ESMDA) [45] was introduced based on the Ensemble Smoother (ES) proposed by van Leeuwen and Evensen [46] in order to avoid stopping and restarting the model when observations happen. Furthermore, a range of methods based on Monte Carlo techniques are formed to conduct DA. For example, the Particle Filter (PF) represents a PDF by ensembles (particles) without the limitation of Gaussianity of the distribution. Bao et. al proposed to combine ESMDA with GAN to deal with the non-Gaussianity [47]. DA algorithms offer an opportunity for optimizing the parameters (i.e. states), quantifying the uncertainty and gradient-free training of ANNs at the same time.

In this paper, a novel training framework for ANNs was proposed by adopting data assimilation. This training framework avoids the dependence of gradient and hence some disadvantages of GD-based methods. ESMDA trains ANNs with pre-defined iterations by updating the parameters using all the available observations which can be regarded as offline learning. EnKF optimizes ANNs when new observation available by updating parameters which can be regarded as real-time learning. To illustrate the idea, a fully connected FNN integrated with EnKF and ESMDA is implemented. Two synthetic cases with the regression problems of Sine function and Mexican Hat function are conducted to test and validate the proposed framework. Furthermore, the uncertainty of FNN parameters is analysed and quantified by ESMDA. The paper is organized as follows. Section 2 provides the theory of FNN, data assimilation and the proposed framework. Section 3 presents the data and settings of the synthetic cases to validate the proposed framework. The results are demonstrated in Sect. 4. Finally, summary and conclusions are given in Sect. 5.

2 Methodology

2.1 Notations

Extensive use of mathematical notations is made in this section. The DA-related notations in this paper are consistent with the symbols used in our previous work [48] as much as possible. To better understand the equations, the notations are summarized and described in advance (Table 1).

2.2 Feedforward neural network

A Feedforward Neural Network (FNN) is an ANN wherein the information flows from the input layer through the transfer functions to the output layer. There are no feedback connections, and hence, the neurons do not form a cycle. Neurons were proposed by Frank Rosenblatt [6] inspired by Warren McCulloch and Walter Pitts [5]. In a neuron, the output is calculated by a nonlinear function (activation function or transfer function) of the sum of its inputs as: $y = f\left(\sum_{i=1}^{p} w_i \psi_i\right)$. An FNN is formed by the combination of such neurons as in the biological neural networks. Without losing generality, the three-layer FNN (input layer, hidden layer and output

Table 1 List of symbols

Symbols	Descriptions
ψ	The inputs of Artificial Neural Networks
w	The weights of Artificial Neural Networks
f	Activation (Transfer) function of Artificial Neural Networks
h	The outputs of hidden layer of Artificial Neural Networks
b	The biases of Artificial Neural Networks
y	The output of output layer of Artificial Neural Networks
n	The number of neurons in the input layer of Artificial Neural Networks
N_h	The number of neurons in the hidden layer of Artificial Neural Networks
m	The number of neurons in the input layer of Artificial Neural Networks
t	Time variable
X	State vectors
A	The parameters vector
B	The state vector
N_x	The dimension of X
N_a	The dimension of A
N_b	The dimension of B
M	Nonlinear model operator
Y	Measurements
H	Observation operator
R	Covariance
P	The covariance matrix of X
N_e	The number of ensembles
K	Kalman gain matrix
α	Inflation coefficient
ξ, ν, ε	Gaussian distribution errors

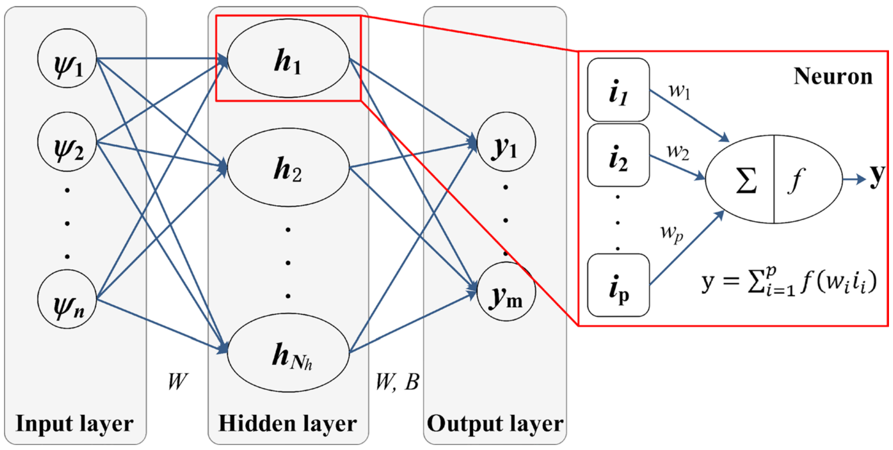

Fig. 1 The structure of Feedforward Neural Networks and neurons

layer) is used as an example of FNNs in this study (Fig. 1). The feedforward process is the same as common fully connected neural networks as follows:

$$h_j = f_1\left(\sum_{i=1}^{n} w_{ji} \times \psi_i\right) \quad i = 1, 2, \ldots, n; j = 1, 2, \ldots, N_h \tag{1}$$

$$y_k = f_2\left(\sum_{j=1}^{N_h} w_{kj} \times h_j + b_j\right) \quad k = 1, 2, \ldots, m \tag{2}$$

where ψ_i, h_j and y_k represent the nodal values in the input layer, hidden layer and output layer, respectively; n, N_h and m are the number of neurons in the input layer, hidden layer and output layer; w_{ji} is the weight connecting the input ψ_i and the jth neuron in the hidden layer; b_j represents the bias in the output layer; w_{kj} is the weight connecting the jth neuron in the hidden layer (h_j) and the output y_k; f_1 and f_2 are the activation functions in the hidden layer and the output layer.

2.3 Data assimilation

Generally, data assimilation combines information from a variety of sources to improve the accuracy of predictions and takes the uncertainty from measurements, inputs, parameters and model structures into account at the same time. In a nonlinear dynamic system, the state space X is defined as:

$$X_t = \begin{pmatrix} A \\ B \end{pmatrix} t = 1, 2, \ldots, N_t \tag{3}$$

where X_t is the state vectors at time t with the dimension of N_x; N_t denotes the number of time steps; A represents the parameters vector with dimension of N_a; B represents other state vectors with dimension of N_b; N_x denotes the number of all state vectors in X_t which equals $N_a + N_b$.

The system is treated as derivations of state equation (Eq. (4)) and observation equation (Eq. (5)) through time t.

$$X_t^f = M_{t-1}\left(X_{t-1}^a\right) + \xi \quad \xi \sim N\left(0, R_\xi\right) \tag{4}$$

$$Y_t = H_{t-1}\left(X_t^f\right) + \nu \quad \nu \sim N\left(0, R_\upsilon\right) \tag{5}$$

where f denotes the forecast (prior estimation) of the states; a denotes the analysis (posterior estimation) of the states; X_t^f represents the forecast of the states at time t; $M_{t\text{-}1}$ is the nonlinear model operator; ${X_{t\text{-}1}}^a$ is the analysis of the states at time t-1; $\xi \sim N\left(0, R_\xi\right)$ and $\nu \sim N\left(0, R_\nu\right)$ indicate the Gaussian distribution errors with zero mean and covariance matrix R_ξ and R_ν; Y_t is the observation vector at time

t; H represents the observation operator which connects the model states and the observations.

2.3.1 Ensemble kalman filter

The EnKF is a sequential data assimilation algorithm based on KF. There are typically two steps in EnKF: the forecast step and the analysis (update) step. In the forecast step, the forecast states is updated according to Eq. (4). In the analysis step, the observation data Y^o are first perturbed by random errors:

$$Y_t^o = Y^o + \varepsilon \quad \varepsilon \sim N\left(0, R_\varepsilon\right) \tag{6}$$

where Y_t^o represents the perturbed observation data at time t; $\varepsilon \sim N(0, R_\varepsilon)$ indicates Gaussian random observation errors with zero mean and covariance matrix R_ε.

The analysis states are obtained by updating the forecast as follows:

$$X_t^a = X_t^f + \frac{P_t^f H^T}{HP_t^f H^T + R_\varepsilon}\left(Y_t^o - Y_t\right) \tag{7}$$

The analysis covariance matrix at time t is

$$P_t^a = \left(I - \frac{P_t^f H^T}{HP_t^f H^T + R_\varepsilon} H\right) P_t^f \tag{8}$$

Here

$$P_t^f = \frac{1}{N_e - 1} \sum_{i=1}^{N_e} \left(x_{i,t}^f - \overline{x_t^f}\right)\left(x_{i,t}^f - \overline{x_t^f}\right)^T \tag{9}$$

$$\overline{x_t^f} = \frac{1}{N_e} \sum_{i=1}^{N_e} x_{i,t}^f \tag{10}$$

Define

$$K_t = \frac{P_t^f H^T}{HP_t^f H^T + R_\varepsilon} \tag{11}$$

where K_t is the Kalman gain matrix at time t; N_e represents the ensemble size; P_t^f is the forecast covariance matrix at time t; $\overline{x_t^f}$ is the mean of ensemble members for forecast states.

Equations (4) ~ (11) illustrate the process of recursive optimal estimation in EnKF which is able to dynamically update the system estimates when new observations become available.

2.3.2 Ensemble smoother with multiple data assimilation

Equations (4)~(11) show that the ensemble-based sequential data assimilation (e.g. EnKF, PF) updates the states at the time when observations happen which results in the necessity of restarting the simulations. The recurrent simulation may be inconvenient when the purpose is to incorporate different kinds of data for history matching. Therefore, Ensemble Smoother with Multiple Data Assimilation (ESMDA) is proposed to update the states by simultaneously assimilating all the available data. Unlike sequential data assimilation, it is not necessary to restart the simulations in ESMDA. This procedure enables ESMDA to obtain better data matches with lower computation cost.

ESMDA is an iterative Ensemble Smoother with a predefined number of iterations for data assimilation. An inflation coefficient α_i is introduced to the measurement error in each iteration. The requirement of inflation coefficient is described in Eq. (12) to maintain correct posterior mean and covariance.

$$\sum_{i=1}^{N_i} \frac{1}{\alpha_i} = 1 \tag{12}$$

where N_i is the predefined number of iterations for data assimilation. Apparently, there are many alternatives for inflation coefficient which satisfies the requirement. The determination of α_i refer to [49].

The inflation coefficient is used to inflate the perturbation of all observation data and its covariance matrix in Eq. (13) and Eq. (14) which leads to:

$$Y = Y^o + \sqrt{\alpha_i}\varepsilon \;\; \varepsilon \sim N\left(0, R_e\right) \tag{13}$$

$$K = \frac{P^f H^T}{HP^f H^T + \sqrt{\alpha_i} R_\varepsilon} \tag{14}$$

2.4 Training FNN with DA

Assume the structure (the number of layers, the nodes in each layer and the connection between nodes) of FNN for a specified problem is determined and represented by M^*. The weights (w in Eq. (1) and (2)) and biases (b in Eq. (2)) are regarded as states (X^*) of M^* which leads to $X^* = \begin{pmatrix} w \\ b \end{pmatrix}$.

In the perspective of DA, substitute M in Eq. (4) with M^*, we can obtain:

$$X_t^{*f} = M_{t-1}^*\left(X_{t-1}^{*a}\right) + \xi^* \;\; \xi \sim N\left(0, R_\xi\right) \tag{15}$$

$$Y_t^* = H^*\left(X_t^{*f}\right) \tag{16}$$

where Y_t^* represents the outputs of M^* with the element of y_k in Eq. (2); X^* is the parameters which can be updated by Eq. (7)–(11).

In the perspective of FNN, the optimization of parameters (w and b) in the back-propagation process is replaced by data assimilation. The ESMDA can be used to train the FNN with the historical data. The sequential data assimilation can be used to adjust the model trained by ESMDA with the real-time observations. The procedure of FNN trained by sequential data assimilation and ESMDA is shown in Fig. 2.

Algorithm 1 Pseudo code of training FNN with EnKF

```
1.  Construct the initial structure of FNN ($M^*$)    // Eq. (1) and Eq. (2)
2.  Initialize the trainable parameters ($\boldsymbol{w}$ and $\boldsymbol{b}$) and the hyper-parameters ($N_h$, $N_e$, $R_\xi$, $R_\varepsilon$)
3.  Generate initial parameter ensembles $X_0^f$ with $N(0, R_\xi)$
4.  FOREACH ($t < N_t$) DO    // For every time step
5.      FOREACH (ens_num < $N_e$) DO    // For every ensemble member
6.          Calculate the hidden layer outputs of FNN    // Eq. (1)
7.          Calculate the output layer outputs of FNN    // Eq. (2)
8.      ENDFOR
9.      Generate the perturbed observations $Y_t^o$ with $N(0, R_\varepsilon)$
10.     Calculate the ensemble mean of forecast states    // Eq. (10)
11.     Calculate the forecast covariance matrix    //Eq. (9)
12.     Update the forecast states and calculate the analysis states    // Eq. (7)
13.     Update the forecast covariance matrix and calculate the analysis covariance matrix
        //Eq. (8)
14.     Assign the analysis states of the current time step as the forecast states of the next time step
15. ENDFOR
```

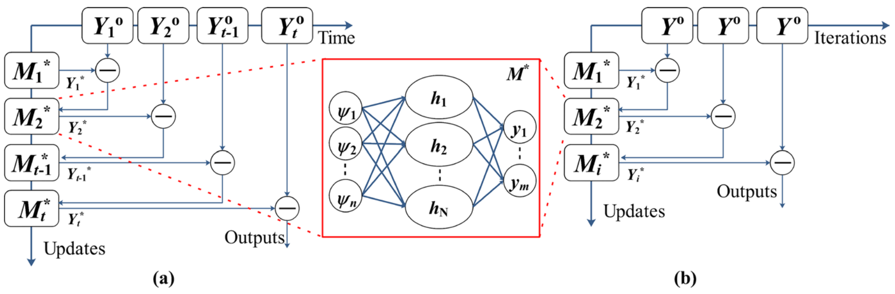

Fig. 2 The procedure of FNN trained by **a** sequential data assimilation; **b** ESMDA. The combination of FNN and DA can be summarized as Algorithm 1 (for EnKF) and Algorithm 2 (for ESMDA). There are several hyper-parameters for Algorithm 1 and Algorithm 2 which should be determined based on the prior information of the actual situation

Algorithm 2 Pseudo code of training FNN with ESMDA

```
1.  Construct the initial structure of FNN ($M^*$)    // Eq. (1) and Eq. (2)
2.  Initialize the trainable parameters ($\boldsymbol{w}$ and $\boldsymbol{b}$) and the hyper-parameters ($N_h$, $N_i$, $N_e$, $R_\xi$, $R_\varepsilon$)
3.  Generate initial parameter ensembles $X_0^f$ with $N(0, R_\xi)$
4.  FOREACH ($iteration < N_i$) DO    // For every iteration
5.      FOREACH($ens\_num<N_e$) DO    // For every ensemble member
6.          Calculate the hidden layer outputs of FNN    // Eq. (1)
7.          Calculate the output layer outputs of FNN    // Eq. (2)
8.      ENDFOR
9.      Generate perturbed observations $Y^o{}_{iteration}$ with $N(0, R_\varepsilon)$
10.     Calculate the ensemble mean of forecast states    // Eq. (10)
11.     Calculate the forecast covariance matrix    //Eq. (9)
12.     Update the forecast states and calculate the analysis states    // Eq. (7) and Eq. (14)
13.     Update the forecast covariance matrix and calculate the analysis covariance matrix
        //Eq. (8) and Eq. (14)
14.     Assign the analysis states of the current time step as the forecast states of the next time step
15. ENDFOR
```

3 Synthetic cases

The performance of the proposed integration of FNN and DA is validated through two synthetic cases. The main purpose of the synthetic cases is to analyse the capability of the proposed method in generating accurate estimations without gradient information by comparing the performance of the proposed method with the traditional GD method. In the synthetic cases, two regression datasets are generated from Sine function and Mexican Hat function (hereafter, refer to Sine function case and Mexican Hat function case). Different optimization methods (GD, EnKF, ESMDA) are conducted to train the FNN model. The methods used in the synthetic cases are summarized in Table 2.

Table 2 Methods used in synthetic cases

Datasets	Model	Optimization of FNN	Performance criteria
Sine function	FNN	GD EnKF ESMDA	RMSE R^2
Mexican Hat function	FNN	GD EnKF ESMDA	RMSE R^2

3.1 Performance criteria

As recommended by [50], the root mean square error (RMSE) and coefficient of determination (R^2) are used to assess the performance of different training algorithms in the two synthetic cases (as shown in Eq. (20) and (21)). The RMSE measures the average magnitude of the error between model simulations (M) and observations (O). As shown in Eq. (20), the errors are squared before averaged, large errors take a relatively high weight. Therefore, RMSE is useful when large errors are undesirable. R^2 measures the predictive ability of models.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(M_i - O_i\right)^2} \tag{20}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(O_i - M_i\right)^2}{\sum_{i=1}^{N}\left(O_i - \overline{O}\right)^2} \tag{21}$$

where N represents the total number of observations; $\overline{O}$ is the average of observations.

Besides, the computation time was also recorded as a criterion to assess the computation costs of different models.

3.2 Data

In the Sine function case, two datasets (training data and validation data) are generated. The data in training stage are generated in (0, 2π) with interval of 0.01π which results in 201 samples. The data in validation stage are generated in (0, 2π) with interval of 0.1 which results in 63 samples. Detail information of the data is summarized in Table 3.

In the Mexican Hat function case, two datasets (training data and validation data) are generated using $\psi(t) = \frac{2}{\sqrt{3\sigma}\pi^{\frac{1}{4}}}\left(1 - \left(\frac{t}{\sigma}\right)^2\right)e^{-\frac{t^2}{2\sigma^2}}$ with $\sigma = 1$. In particular, 200 samples are generated in [− 5, 5] for the training dataset; 30 samples are generated in [− 5, 5] for the validation dataset. Detail information of the data is summarized in Table 4.

Table 3 Data description for the Sine function case

	Training stage			Validation stage		
	Number of samples	Range	Interval	Number of samples	Range	Interval
Input	201	[0, 2π]	0.01π	63	[0, 2π]	0.1
Output	201	[− 1, 1]	*	63	[− 1, 1]	*

Table 4 Data description for the Mexican Hat function case

	Training stage		Validation stage	
	Number of samples	Range	Number of samples	Range
Input	200	$[-5, 5]$	30	$[-5, 5]$
Output	200	$\left[-\frac{4}{\sqrt{3}\pi^{\frac{1}{4}}}e^{-\frac{3}{2}}, \frac{2}{\sqrt{3}\pi^{\frac{1}{4}}}\right]$	30	$\left[-\frac{4}{\sqrt{3}\pi^{\frac{1}{4}}}e^{-\frac{3}{2}}, \frac{2}{\sqrt{3}\pi^{\frac{1}{4}}}\right]$

The data used in the two synthetic cases are shown in Fig. 3.

3.3 Experimental Settings

The architecture of FNN is predefined to be fully connected network with one input layer, one hidden layer and one output layer. Based on the features of the dataset, the number of neurons in each layer is one, ten and one. respectively. The parameters (weights and bias) are randomly initialized from a normal distribution. The loss function in gradient decent method is mean square error (MSE). 10,000 epochs with learning rate of 0.12 are used for gradient decent method to train the FNN. Without loss of generality, the biases between the hidden layer and output layer are selected as states to be updated in EnKF. The hyperparameters in EnKF, the ensemble size N_e, the prior parameter covariance matrix R_ξ and the observation error covariance matrix R_ε are 50, 0.1 and 0.005, respectively. The observation error covariance R_ε is set to be a small value because the observations used in EnKF are generated from the Sine wave which was accurate and much more trustworthy than the FNN model. EnKF is used in the ESMDA to conduct the procedure of data assimilation. The predefined number of iterations for data assimilation N_i, the ensemble size N_e, the prior

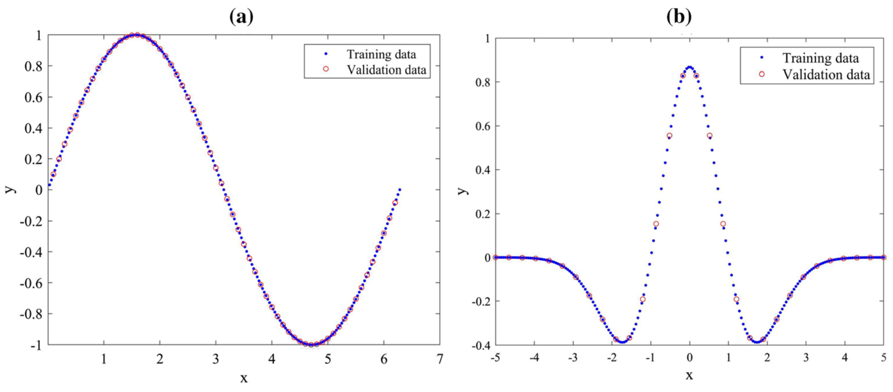

Fig. 3 Data used in the training stage and validation stage for **a** the Sine function case and **b** the Mexican Hat function case

parameter covariance matrix R_ξ and the observation error covariance matrix R_ϵ are 3, 50, 0.1 and 0.1, respectively.

4 Results and discussions

4.1 Performance of FNN model optimized by EnKF

In the Sine function case, the results calculated from FNN which optimized by different methods are shown in Fig. 4. After 10,000 epochs of training, the FNN model with gradient decent method approaches the Sine Function with some biases. The RMSE and R^2 values for GD-optimized FNN model are 0.0948 and 0.9819. Although the values of performance criteria are relatively acceptable, there is still bias in the peak and trough of the curve which may be caused by the difference of gradient changes and static learning rate of the algorithm. In the experiment of EnKF-optimized FNN model, there are ensembles for the parameters which generated from a random normal distribution. To calculate the performance criteria, the ensemble mean is used as the final model outputs. The EnKF-optimized FNN model indicates a better match to the Sine Function (the red curve in Fig. 4) with RMSE of 0.0317 and R^2 of 0.9980. Each realizations of parameters can be regarded as a possible realization of FNN. On the contrary to the gradient decent algorithm, EnKF is capable of capturing the variance of gradient changes because of the updating procedure in the algorithm. The evolution of parameters (shown in Fig. 5) also reflects the correction processes of the parameters to adapt the larger gradient changes. After randomly generating the FNN parameters (biases from the hidden layer to the output layer), the uncertainty of parameter remains relatively large because of the large difference between the FNN model and the Sine wave according to Eq. (7). The

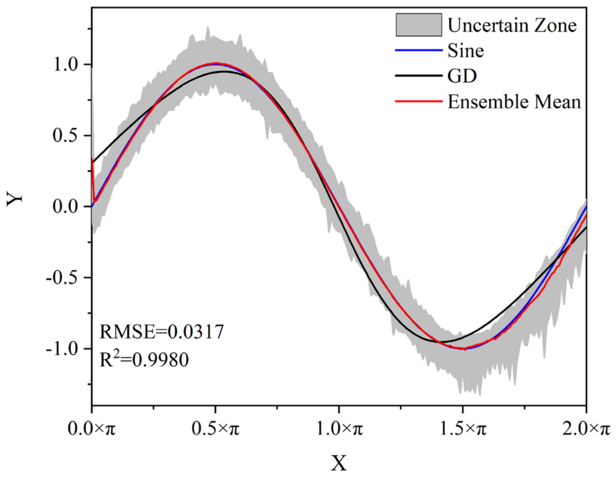

Fig. 4 Comparison of the results from FNN optimized by Gradient Descent (black curve) and EnKF (Red curve for ensemble mean and grey area for uncertain zone) in the Sine function case (color figure online)

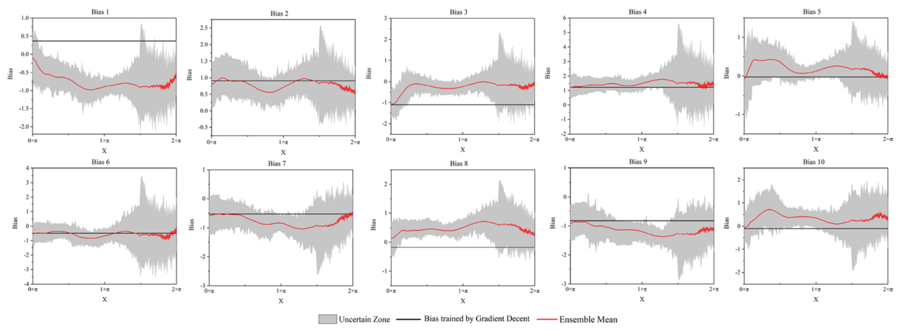

Fig. 5 Parameters trained by EnKF and Gradient Decent in the Sine function case

same situation can be found at "$x = 1.5\pi$". On the contrary, the parameter uncertainty is reduced when "$x \in (0.75\pi, 1.25\pi)$" because of the relatively small difference between the FNN model and the Sine wave (Fig. 5). These results indicate that the EnKF optimized FNN model with higher accuracy than gradient decent algorithm. Furthermore, the EnKF is able to optimize the parameters of FNN in real time by incorporating real-time observations which is intrinsic quality of the methods.

In the Mexican Hat function case, the hyper-parameters of FNN and EnKF are identical with those in the Sine function case. The results calculated from FNN which optimized by different methods are shown in Fig. 6. The RMSE and R^2 value for GD-optimized FNN model are 0.0329 and 0.9891. In the experiment of EnKF-optimized FNN, the ensemble means of the model outputs are used to calculate the performance criteria with RMSE of 0.018 and R^2 of 0.9967. The better

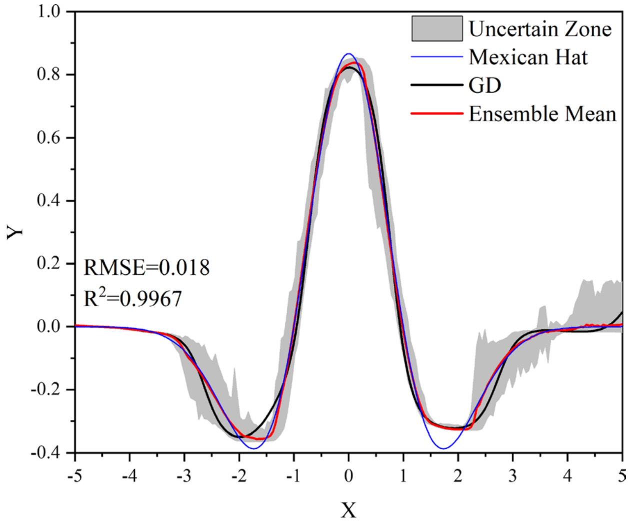

Fig. 6 Comparison of the results from FNN optimized by Gradient Descent (black curve) and EnKF (Red curve for Ensemble mean and grey area for uncertain zone) in the Mexican Hat function case (color figure online)

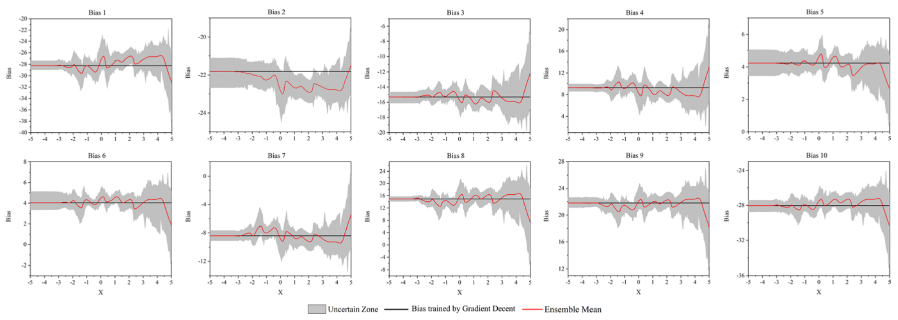

Fig. 7 Parameters trained by EnKF and Gradient Decent in the Mexican Hat function case

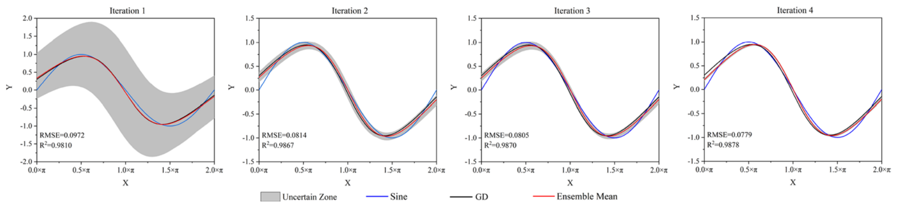

Fig. 8 Comparison of the results from FNN optimized by Gradient Descent (black curve) and. ESMDA (Red curve for Ensemble mean and grey area for each ensemble) in the Sine function case (color figure online)

performance is attributed to the update scheme of the model states which is shown in Eq. (7) ~ (11). The evolution of parameters shown in Fig. 7 illustrates the update process. From Fig. 6 and Fig. 7, one can tell that the variance of parameters is larger when the difference between observations (Y_t^o) and simulations (Y_t) are large which can be explained by Eq. (7). These results indicate that EnKF is able to optimize the parameters of FNN by implementing the update process with higher accuracy.

4.2 Performance of FNN model optimized by ESMDA

In the Sine function case, the outputs and the corresponding parameters of four iterations are shown in Figs. 8 and 9. Figure 8 displays the outputs of the models with grey area indicating uncertain zone, blue curve indicating Sine function, black curve indicating optimized outputs from GD, red curve indicating ensemble means of the outputs. In the first iteration, 50 samples of parameters are randomly generated using normal distribution with covariance matrix R_ξ, the FNN model is executed with the generated samples to yield outputs. The uncertain zone of the outputs for the first iteration is the largest because of the random generation of parameters (shown in Fig. 9) which results in the largest uncertainty of the parameters. In the second iteration, the distributions of the parameters are updated by the EnKF which significantly narrows down the uncertain zone of the outputs. In the third and fourth iteration, the distributions of the parameters are slightly updated without significant effects on

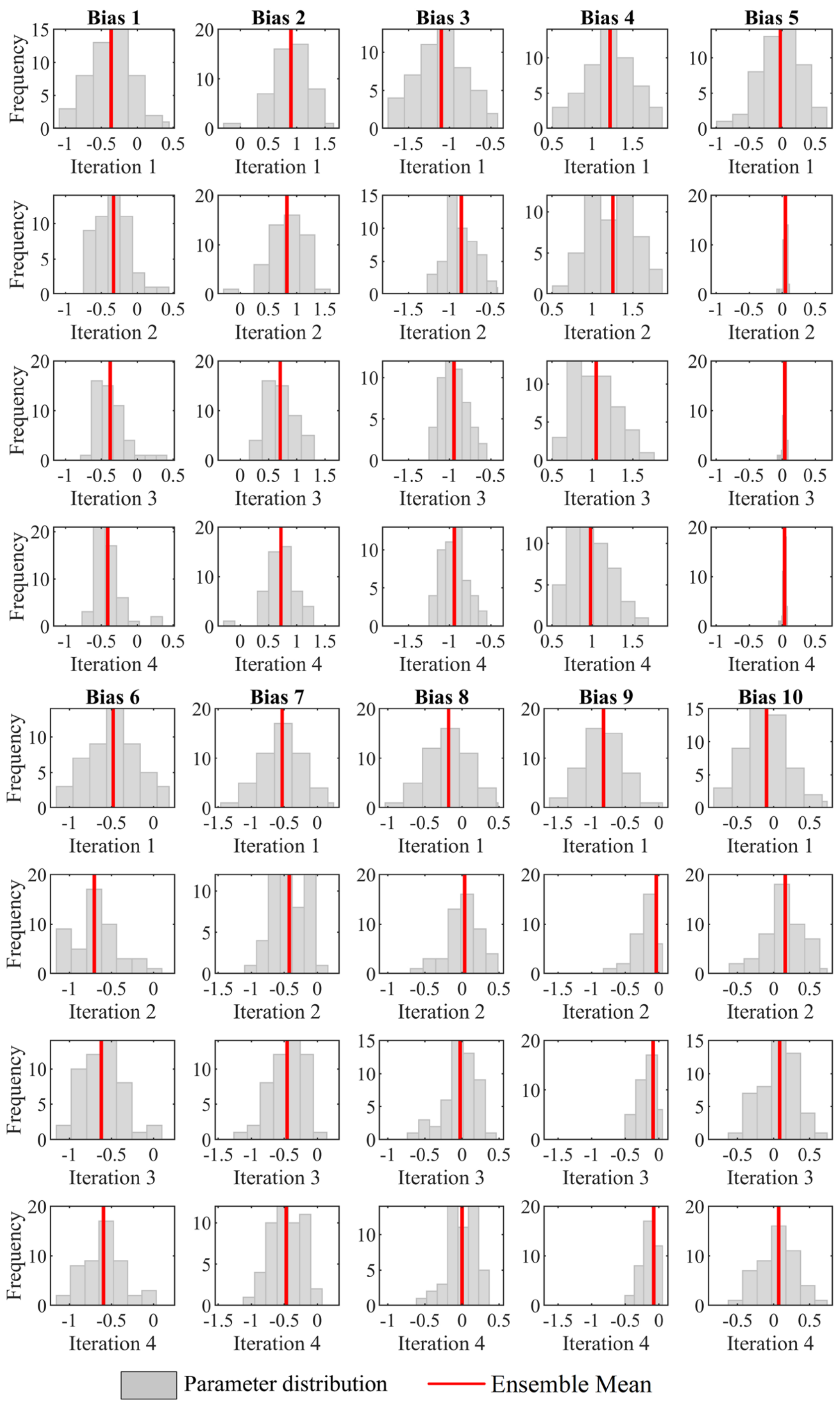

Fig. 9 Parameters trained by ESMDA in the Sine function case

Table 5 RMSE and R^2 values for the Gradient Decent and ESMDA methods in the Sine function case

	Gradient Decent	ESMDA			
		Iteration 1	Iteration 2	Iteration 3	Iteration 4
RMSE	0.0948	0.0972	0.0814	0.0805	0.0779
R^2	0.9819	0.9810	0.9867	0.9870	0.9878

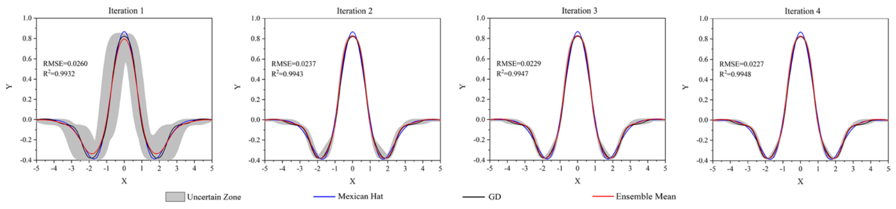

Fig. 10 Comparison of the results from FNN optimized by Gradient Descent (black curve) and ESMDA (Red curve for Ensemble mean and grey area for uncertain zone) in the Mexican Hat function case (color figure online)

the outputs. The mean of the 50 ensembles is considered as the best estimation for the outputs in each iteration. The RMSE and R^2 are calculated to conduct quantitative comparisons between the observations and simulations (Table 5). Table 5 indicates that better results are obtained by ESMDA than those obtained by GD which proves the effectiveness of the ESMDA for updating the parameters. The evolution of parameters (biases from the hidden layer to the output layer) are shown in Fig. 9. In each figure, the histogram of the ensembles is used to indicate the distribution of the parameters. The red line indicates the ensemble mean of the updated parameters. The convergence of the parameters with the increase of iterations indicates the effectiveness of the ESMDA. The variances of the parameters decrease with the iterations which could be obtained from Fig. 9 by the narrowing of the uncertain zone. The mean value of the parameter in Fig. 9 which corresponds to the mean value of the trained results in Fig. 8 can be regarded as the optimal parameters for the FNN model.

In the Mexican Hat function case, the outputs and the corresponding parameters of four iterations are shown in Figs. 10 and 11, respectively. Similar to Figs. 8, 10 shows the outputs of the models with grey area indicating uncertain zone, blue curve indicating Sine function, black curve indicating optimized outputs from GD, red curve indicating ensemble means of the outputs. The uncertain zone of the outputs for the first iteration is the largest because of the random generation of parameters (shown in Fig. 11). In iteration 2 and iteration 3, the uncertain zone of the outputs keeps narrowing down due to the parameters updating process of ESMDA. It should be noted that the uncertainties of parameters were larger when the gradient of Mexican Hat function closing zero (i.e. around $x = \pm 3$, $x = \pm\sqrt{3}$ and $x = 0$). The differences between the outputs of FNN and the Mexican Hat function are also relatively larger at these points. The reason may also lie in Eq. (7) as we described in Sect. 4.1.

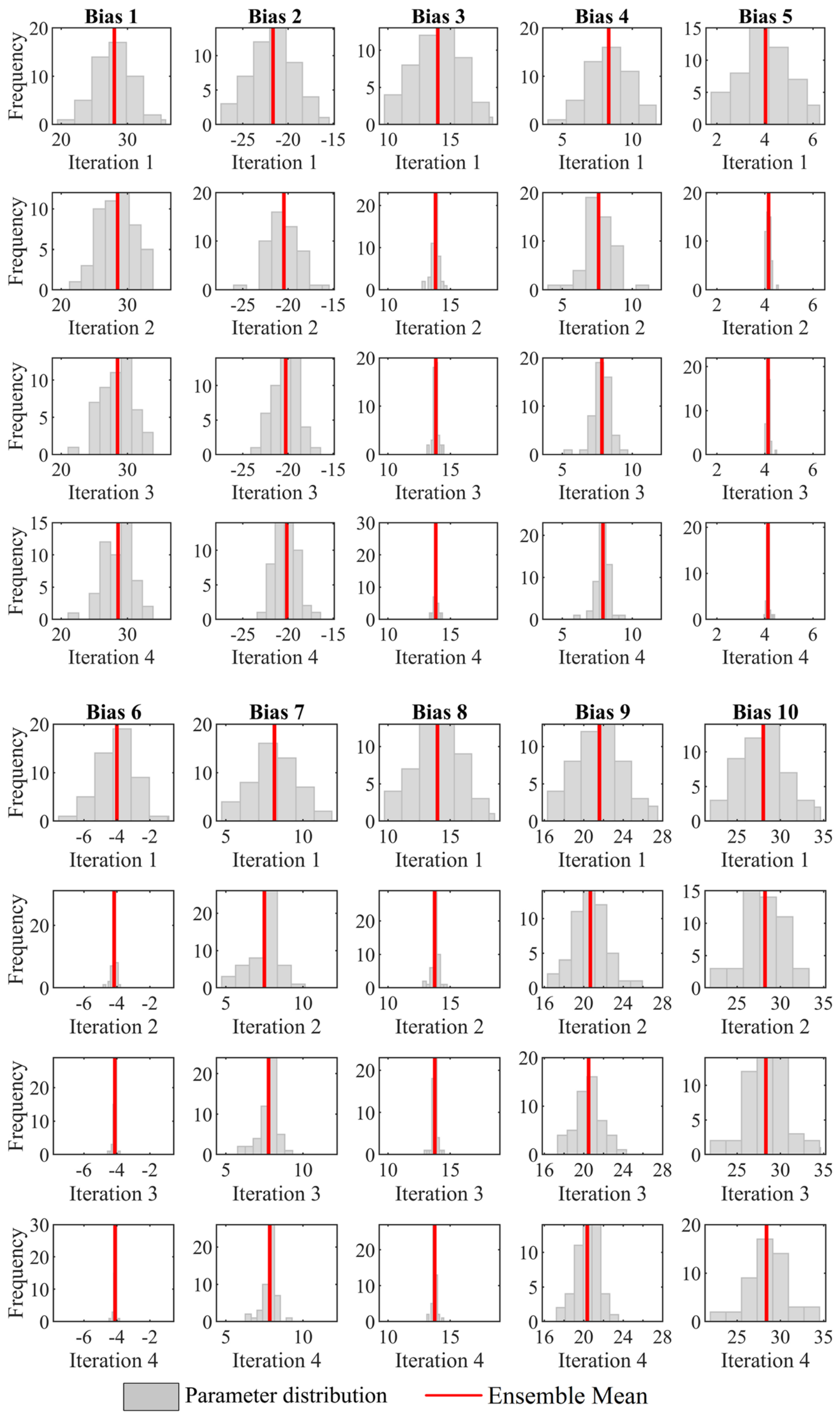

Fig. 11 Parameters trained by ESMDA and Gradient Decent in the Mexican Hat function case

This phenomenon indicates the adjustment of parameters (shown in Fig. 11) according to the observations which also demonstrates the effectiveness of updating processes in ESMDA. It should also be noted that the variance of parameters is not enough to cover some points in the model outputs (i.e. $\pm\sqrt{3}$ in Fig. 10). This may be caused by the situation that only biases in the hidden layer are perturbed and updated. Involving more parameters (for instance, weights in Eq. (1) ~ (2)) for perturbation and optimization may solve this problem. Quantitative comparisons between the observations and simulations are conducted by calculating RMSE and R^2 (Table 6). Table 6 shows that better results are obtained by ESMDA than those obtained by Gradient Decent. The evolution of parameters is shown in Fig. 11. In each figure, the histogram of the ensembles is used to indicate the distribution of the parameters. The red line indicates the ensemble mean of the updated parameters. The variances of parameters are lowered with the iterations which results in the narrowing of uncertain zones in Fig. 10.

4.3 Validation

The FNN trained by GD and ESMDA are then validated using the validation data generated in Sect. 3.3. The ensemble means of ESMDA parameters are used as optimal parameters. The EnKF trained FNN is not validated due to two reasons. The first reason is that EnKF is used for real-time training (online learning) in the proposed training framework which would conduct continuous learning when the new observations available. The second reason is that EnKF optimizes the parameters based on the observations for a particular time step. The validation results for the Sine function case and the Mexican Hat function case are illustrated in Fig. 12 which shows considerable match in both cases. Table 7 shows the RMSE and R^2 values for the Gradient Decent and ESMDA methods in validation stage which indicates a reasonable training by comparing with Tables 5 and 6. The generalization ability of GD and ESMDA is both acceptable. The performance of ESMDA is slightly better than GD.

4.4 Uncertainty analysis

Uncertainty is inevitable in all kinds of prediction models including neural networks. Due to the nature of EnKF, the uncertainty of FNN parameters (i.e. biases) is quantified through the generation of ensembles for the parameters. There are many ways to describe parameters uncertainty via ensemble analysis, including histograms, probability, ensemble means, standard deviations (STD) and correlations. Figures 9 and 11 are examples which show the changes of uncertainty of the parameters over different iterations by histograms. "Iteration 1" denotes the initial step of the ESMDA which randomly generates the parameters from a normal distribution. The ensemble spread of the parameters decreases after the first iteration and continues to be narrowed as the ESMDA executes. The narrowing of the range of uncertainty is observed for many, but not all, of the parameters.

Table 6 RMSE and R^2 values for the Gradient Decent and ESMDA methods in the Mexican Hat function case

	Gradient Decent	ESMDA			
		Iteration 1	Iteration 2	Iteration 3	Iteration 4
RMSE	0.0329	0.0260	0.0237	0.0229	0.0227
R^2	0.9891	0.9932	0.9943	0.9947	0.9948

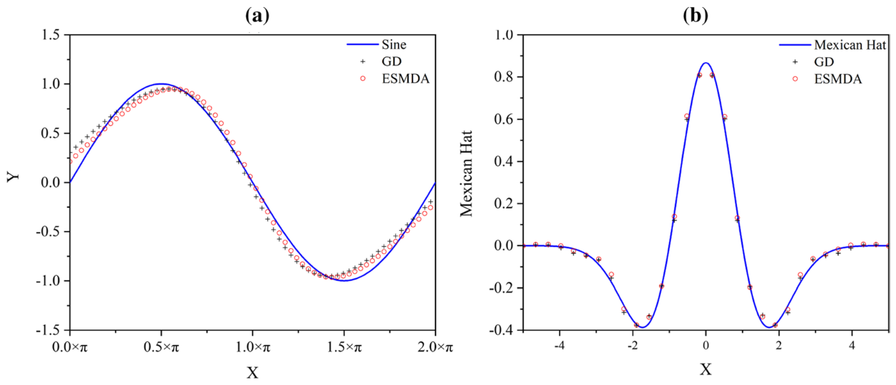

Fig. 12 The validation of the Gradient Decent and ESMDA trained FNN for: **a** the Sine function case; **b** The Mexican Hat function case

Table 7 RMSE and $R.^2$ values for the Gradient Decent and ESMDA methods in validation stage

	Sine function		Mexican Hat function	
	Gradient Decent	ESMDA	Gradient Decent	ESMDA
RMSE	0.0969	0.0777	0.0228	0.0226
R^2	0.9812	0.9879	0.9946	0.9947

A quantitative summary of the uncertainty is provided which summarizes the ensemble maximums, ensemble minimums, ensemble means and STDs for all the parameters in the Sine function case (Table 8) and Mexican Hat function case (Table 9). It should be noted that statistical values are used because of the existence of ensembles in ESMDA. Different parameters are listed in columns. The values of ensemble maximums, ensemble minimums, ensemble means and STDs in different iterations are demonstrated in different rows. In both cases, the ensemble STDs for all the parameters narrow along with the iterations which indicates a reduction in the uncertainty. However, different degrees of narrowing are observed for different parameters. The narrowing for "Bias 5" in the Sine function case and "Bias 3 ~ Bias 8" in the Mexican Hat function case indicates a significant reduction in uncertainty and reveals that these particular parameters identifiable from the data (also shown in Figs. 9 and 11). For the other

Table 8 Statistical values of FNN parameters from ESMDA in the Sine function case

	Parameters	Bias 1	Bias 2	Bias 3	Bias 4	Bias 5	Bias 6	Bias 7	Bias 8	Bias 9	Bias 10
Maximum	Iteration 1	0.44	1.66	−0.41	1.87	0.67	0.19	0.24	0.49	0.05	0.74
	Iteration 2	0.37	1.45	−0.42	1.95	0.09	−0.05	0.16	0.53	0.65	0.92
	Iteration 3	0.28	1.18	−0.59	1.72	0.07	0	0.13	0.35	0.57	0.77
	Iteration 4	0.26	1.2	−0.59	1.65	0.07	0.03	0.05	0.36	0.52	0.75
Minimum	Iteration 1	−1.09	−0.3	−1.75	0.5	−0.99	−1.16	−1.46	−1.04	−1.63	−0.83
	Iteration 2	−0.73	−0.24	−1.2	0.67	−0.07	−1.2	−1.09	−0.54	−0.63	−0.47
	Iteration 3	−0.68	−0.37	−1.25	0.54	−0.06	−1.17	−1.13	−0.62	−0.44	−0.54
	Iteration 4	−0.72	−0.16	−1.22	0.52	−0.05	−1.09	−1.1	−0.57	−0.39	−0.54
Mean	Iteration 1	−0.37	0.9	−1.1	1.22	−0.03	−0.48	−0.53	−0.18	−0.82	−0.1
	Iteration 2	−0.33	0.83	−0.86	1.25	0.05	−0.7	−0.42	0.03	−0.04	0.16
	Iteration 3	−0.38	0.71	−0.95	1.05	0.04	−0.62	−0.46	−0.03	−0.09	0.08
	Iteration 4	−0.41	0.72	−0.94	0.98	0.04	−0.6	−0.47	0	−0.08	0.07
STD	Iteration 1	0.32	0.34	0.31	0.31	0.32	0.31	0.32	0.31	0.32	0.32
	Iteration 2	0.22	0.29	0.19	0.3	0.03	0.28	0.27	0.24	0.24	0.28
	Iteration 3	0.2	0.28	0.15	0.28	0.02	0.25	0.26	0.22	0.21	0.27
	Iteration 4	0.19	0.25	0.15	0.26	0.02	0.25	0.25	0.2	0.19	0.27

Table 9 Statistical values of FNN parameters from ESMDA in the Mexican Hat function case

	Parameters	Bias 1	Bias 2	Bias 3	Bias 4	Bias 5	Bias 6	Bias 7	Bias 8	Bias 9	Bias 10
Maximum	Iteration 1	35.76	−15.46	18.38	11.71	6.28	−0.86	11.87	18.57	27.49	34.63
	Iteration 2	33.48	−15.89	14.49	10.30	4.50	−3.82	9.98	14.70	24.93	34.81
	Iteration 3	32.73	−16.41	14.36	9.46	4.43	−3.87	9.16	14.47	24.02	34.56
	Iteration 4	32.76	−16.74	14.29	9.18	4.38	−3.90	8.93	14.37	23.44	34.48
Minimum	Iteration 1	19.46	−27.40	9.72	3.97	1.76	−7.53	4.74	9.73	16.33	21.93
	Iteration 2	21.71	−24.80	12.88	4.62	4.02	−4.71	4.84	12.83	17.04	22.01
	Iteration 3	22.21	−23.93	13.21	5.60	4.01	−4.54	5.95	13.13	17.68	22.41
	Iteration 4	22.55	−23.40	13.32	6.08	4.01	−4.44	6.48	13.23	17.98	22.66
Mean	Iteration 1	28.03	−21.65	13.99	8.32	4.02	−4.00	8.15	13.98	21.60	28.04
	Iteration 2	28.52	−20.45	13.81	7.59	4.15	−4.17	7.50	13.75	20.67	28.23
	Iteration 3	28.50	−20.25	13.83	7.83	4.13	−4.13	7.77	13.76	20.49	28.34
	Iteration 4	28.59	−20.13	13.83	7.91	4.13	−4.11	7.85	13.76	20.36	28.41
STD	Iteration 1	3.01	2.50	2.01	1.57	1.06	1.17	1.54	1.98	2.50	2.87
	Iteration 2	2.59	1.72	0.30	1.00	0.09	0.14	1.07	0.30	1.70	2.53
	Iteration 3	2.35	1.40	0.21	0.62	0.07	0.09	0.62	0.21	1.37	2.41
	Iteration 4	2.23	1.21	0.17	0.49	0.06	0.07	0.47	0.17	1.18	2.34

parameters in the Sine function case and the Mexican Hat function case, the ensemble spreads narrow a little over the three iterations which indicates that the data contains relatively little information about the uncertain parameters and the parameters are less identifiable. Tables 8 and 9 indicate that for most parameters the largest uncertainty decrease happens with the first iteration (from "Iteration 1" to "Iteration 2"). Corresponding to this phenomenon, the uncertain zone of outputs from FNN model narrows the most in the first iteration (shown in Figs. 8 and 10). On the one hand, this suggests that the capability of ESMDA to update the parameters uncertainty. On the other hand, this also suggests that even though the nonlinearity exists, the overall relationship between the parameters and data is relatively linear. The ensemble means over different iterations in Tables 8 and 9 indicate the capability of ESMDA to update the value of parameters. Comparing the ensemble means in the Sine function case (Table 8) with the ensemble means in the Mexican Hat function case, we can find that the degree of updating ensemble means in the Sine function case is bigger in the Mexican Hat function case. The reason may be the relatively optimal parameters are obtained in the prior estimates in the Mexican Hat function case. This phenomenon is also revealed in the smaller degree of updating of STDs in the Sine function case than in the Mexican Hat function case.

Furthermore, a comparison of computation cost is conducted to approximately se the complexity of different training methods (GD, EnKF and ESMDA). Computation times (Table 10) under the same software and hardware runtime environment are recorded to indicate the computation cost. The computation cost of ESMDA is similar to GD. The EnKF is much more computational expensive than GD and ESMDA which indicates the fact that the major drawback of EnKF is computation cost. This drawback is reasonable given the fact that EnKF derived from the merge of Kalman Filter [30] and Monte Carlo estimation methods [31]. Therefore, EnKF restarts and executes FNN N_e (ensemble size) times at each time step when observations become available. In our synthetic cases, the number of time steps for observations (N_t) are 201 for the Sine function case and 200 for the Mexican Hat function case. The FNN is executed $N_e \times N_t$ times (50×201 or 50×200 in the synthetic cases) in EnKF which may consume a lot computation time. However, in the proposed training framework, EnKF is used for real-time training (online learning) which means only N_e times executions of FNN are needed when the observations become available. This process involves new information from observations and avoids the retraining of FNN.

Table 10 The computation time of different training methods (GD, EnKF, ESMDA) in the Sine function case and the Mexican Hat function case

	The sine function case			Mexican hat function case		
	GD	EnKF	ESMDA	GD	EnKF	ESMDA
Training time (s)	6.38	41.80	6.33	6.59	42.96	6.28

5 Conclusion

In this paper, a new training framework for neural networks based on data assimilation is proposed to avoid the calculation of gradient in the neural network training. The Feedforward Neural Networks (FNNs), Ensemble Kalman Filter (EnKF) and Ensemble Smoother with Multiple Data Assimilation (ESMDA) are used to validate the proposed framework. Synthetic cases with data generated from the Sine function and the Mexican Hat function are implemented to test the methods. EnKF updates the parameters when the observations available which can be regarded as real-time training (online learning). ESMDA updates the parameters using all the available observations with a predefined number of iterations for data assimilation which can be regarded as normal training (offline learning) compared to the conventional methods. The results from EnKF-optimized and ESMDA-optimized FNN model show higher accuracy than those from gradient-decent-optimized FNN model. This indicates the effectiveness of the EnKF and ESMDA trained FNN. Furthermore, the uncertainty of the FNN model parameters is quantified at the same time. The major advantages of the proposed training methods based on the data assimilation were (1) the avoidance of calculating gradient, (2) the ability of real-time training when the observations available, (3) the uncertainty analysis for the parameters of neural networks. Although only FNN, EnKF and ESMDA were implemented as examples in this study, the potential of data assimilation algorithms on training neural networks is unlimited. Future works may include exploring new data assimilation algorithms (e.g. Particle Filter), exploring other kinds of neural networks (e.g. Recurrent Neural Network, Graph Neural Networks), involving more parameters of neural networks and validating the methods with real observation data.

Acknowledgements The authors would like to thank the reader for their precious time and efforts and providing constructive comments (if possible) to improve the paper. This work was supported by the National Natural Science Foundation of China [grant No. 62006247]; PetroChina Innovation Foundation [2020D-5007-0301]; the National Key R&D Program of China [grant No. 2019YFC1510501].

Data availability All data generated or analysed during this study are included in this published article.

References

1. Huang X, Gao L, Crosbie RS, Zhang N, Fu GB, Doble R (2019) Groundwater recharge prediction using linear regression. Multi-Layer Perception Network, and Deep Learning, Water 11:19
2. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444
3. Jin KH, McCann MT, Froustey E, Unser M (2017) Deep convolutional neural network for inverse problems in imaging. IEEE Trans Image Process 26:4509–4522
4. Zhou H, Chen C, Liu H, Qin F, (2019) Liang H Proactive Knowledge-Goals Dialogue System Based on Pointer Network, CCF International Conference on Natural Language Processing and Chinese Computing, (Springer), pp. 724–735.
5. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5:115–133

6. Rosenblatt F, The perceptron, a perceiving and recognizing automaton Project Para (Cornell Aeronautical Laboratory, 1957).
7. Hopfield JJ (1982) Neural networks and physical systems with emergent collective computational abilities. Proc Natl Acad Sci USA 79:2554–2558
8. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323:533–536
9. Cybenko G (1989) Approximation by superpositions of a sigmoidal function. Math Control Signals Systems 2:303–314
10. Hornik K (1991) Approximation capabilities of multilayer feedforward networks. Neural Netw 4:251–257
11. Zhao H (2016) General vector machine, arXiv preprint
12. Khan NA, Hameed T, Razzaq OA, Ayaz M (2019) Tracking the chaotic behaviour of fractional-order Chua's system by Mexican hat wavelet-based artificial neural network. J Low Freq Noise Vib Act Control 38:1279–1296
13. Gray A, Wimbush A, de Angelis M, Hristov PO, Calleja D, Miralles-Dolz E, Rocchetta R (2022) From inference to design: A comprehensive framework for uncertainty quantification in engineering with limited information. Mech Syst Signal Process 165:108210
14. Song X, Zhan C, Kong F, Xia J (2011) Advances in the study of uncertainty quantification of large-scale hydrological modeling system. J Geog Sci 21:801
15. Her Y, Yoo S-H, Cho J, Hwang S, Jeong J, Seong C (2019) Uncertainty in hydrological analysis of climate change: multi- parameter vs. multi-GCM ensemble predictions, Scientific Reports, p 9.
16. Beven K (2006) On undermining the science? Hydrol Process 20:3141–3146
17. Li L, Xia J, Xu C-Y, Singh VP (2010) Evaluation of the subjective factors of the GLUE method and comparison with the formal Bayesian method in uncertainty assessment of hydrological models. J Hydrol 390:210–221
18. Engeland K, Xu C-Y, Gottschalk L (2005) Assessing uncertainties in a conceptual water balance model using Bayesian methodology / Estimation bayésienne des incertitudes au sein d'une modélisation conceptuelle de bilan hydrologique, Hydrological Sciences Journal, 50 null-63.
19. Beven K, Freer J (2001) Equifinality, data assimilation, and uncertainty estimation in mechanistic modelling of complex environmental systems using the GLUE methodology. J Hydrol 249:11–29
20. Daley R, Atmospheric Data Analysis (Cambridge University Press, 1993)
21. Houtekamer PL, Zhang FQ (2016) Review of the ensemble kalman filter for atmospheric data assimilation. Mon Weather Rev 144:4489–4532
22. Tribbia J, Baumhefner D (2004) Scale interactions and atmospheric predictability: an updated perspective. Mon Weather Rev 132:703–713
23. Leith C (1993) Numerical models of weather and climate. Plasma Phys Controlled Fusion 35:919
24. Wunsch C, Discrete inverse and state estimation problems: with geophysical fluid applications (Cambridge University Press, 2006).
25. Biegler LT, Coleman TF, Conn AR, Santosa FN, Large-Scale Optimization with Applications: Part I: Optimization in Inverse Problems and Design (Springer Science & Business Media, 2012).
26. Emerick A, History Matching and Uncertainty Characterization: Using Ensemble-based Methods (LAP LAMBERT Academic Publishing, 2012).
27. C.D. Rodgers, Inverse methods for atmospheric sounding: theory and practice (World scientific, 2000).
28. A. Tarantola, Inverse problem theory and methods for model parameter estimation (siam, 2005).
29. Evensen G (1994) Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. J Geophys Res 99:10–10
30. Kalman RE (1960) A new approach to linear filtering and prediction problems. J Fluids Eng 82:35–45
31. Kalman RE, Bucy RS (1961) New results in linear filtering and prediction theory. J Basic Eng 83:95–108
32. Aanonsen SI, Nævdal G, Oliver DS, Reynolds AC, Vallés B (2009) Review of ensemble Kalman filter in petroleum engineering. Spe J 14:393–412
33. Hendricks Franssen HJ, Kinzelbach W (2008) Real-time groundwater flow modeling with the Ensemble Kalman Filter: Joint estimation of states and parameters and the filter inbreeding problem, Water Resour Res, p 44

34. Erazo DEG, Wallscheid O, Bocker J (2020) Improved fusion of permanent magnet temperature estimation techniques for synchronous motors using a kalman filter. IEEE Trans Ind Electron 67:1708–1717
35. Bocquet M, Brajard J, Carrassi A, Bertino L (2020) Bayesian inference of chaotic dynamics by merging data assimilation, machine learning and expectation-maximization, Foundations of Data. Science 2:55–80
36. Nóbrega JP, Oliveira ALI (2019) A sequential learning method with Kalman filter and extreme learning machine for regression and time series forecasting. Neurocomputing 337:235–250
37. Mu T, Nandi AK (2009) Automatic tuning of L2-SVM parameters employing the extended Kalman filter. Expert Syst 26:160–175
38. Shah S, Palmieri F, Datum M (1992) Optimal filtering algorithms for fast learning in feedforward neural networks. Neural Netw 5:779–787
39. Sum J, Chi-Sing L, Young GH, Wing-Kay K (1999) On the Kalman filtering method in neural network training and pruning. IEEE Trans Neural Networks 10:161–166
40. Youmin Z, Li XR (1999) A fast U-D factorization-based learning algorithm with applications to nonlinear system modeling and identification. IEEE Trans Neural Networks 10:930–938
41. Simon D (2002) Training radial basis neural networks with the extended Kalman filter. Neurocomputing 48:455–475
42. Obradovic D (1996) On-line training of recurrent neural networks with continuous topology adaptation. IEEE Trans Neural Networks 7:222–228
43. Puskorius GV, Feldkamp LA (1994) Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks. IEEE Trans Neural Networks 5:279–297
44. Chen Y, Chang H, Meng J, Zhang D (2019) Ensemble Neural Networks (ENN): a gradient-free stochastic method. Neural Netw 110:170–185
45. Emerick AA, Reynolds AC (2013) Ensemble smoother with multiple data assimilation. Comput Geosci 55:3–15
46. Leeuwen PJv, Evensen G (1996) Data assimilation and inverse methods in terms of a probabilistic formulation. Mon Weather Rev 124:2898–2913
47. Bao J, Li L, Redoloza F (2020) Coupling ensemble smoother and deep learning with generative adversarial networks to deal with non-Gaussianity in flow and transport data assimilation. J Hydrol 590:125443
48. Li Y, Chen C, Zhou J, Zhang G, Chen X (2016) Dual state-parameter simultaneous estimation using localised Ensemble Kalman Filter and application in environmental model. Int J Embedded Syst 8:93–103
49. Emerick AA, Reynolds AC (2012) History matching time-lapse seismic data using the ensemble Kalman filter with multiple data assimilations. Computat Geosci 16:639–659
50. Moriasi DN (2007) Model evaluation guidelines for systematic quantification of accuracy in watershed simulations. Trans ASABE 50:885–880

Authors and Affiliations

Chong Chen[1] · **Yixuan Dou**[1] · **Jie Chen**[1] · **Yaru Xue**[1]

[1] College of Information Science and Engineering, China University of Petroleum-Beijing, Beijing 102249, China